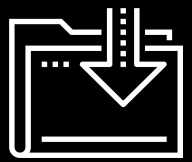
GeoJSON & Leaflet Plugins

Data Boot Camp

Lesson 17.2



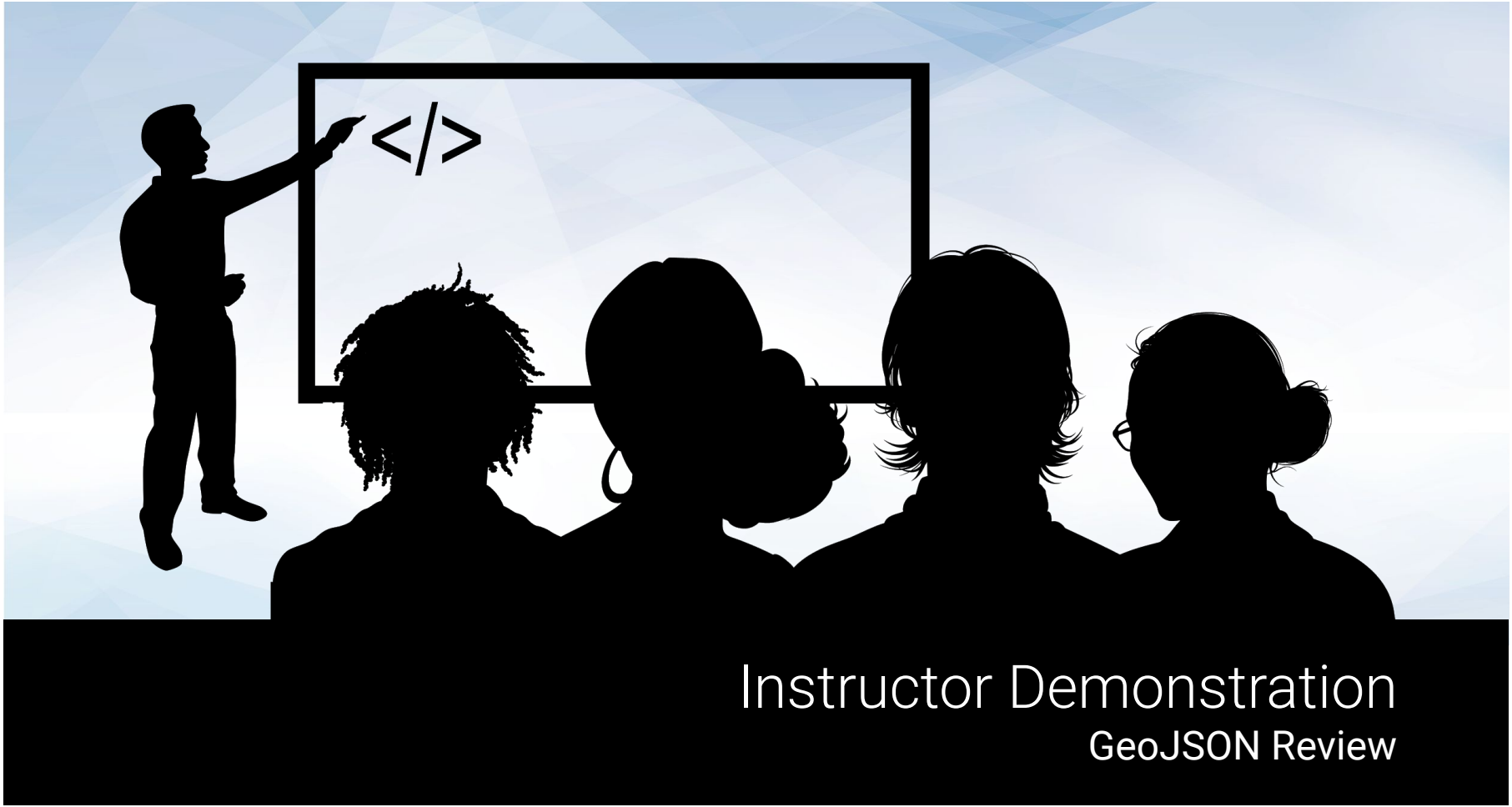
Class Objectives

By the end of today's class you will be able to:

Gain a firm grasp of mapping with GeoJSON.

Learn about and practice using Leaflet plugins and third-party libraries.

Learn how different maps can effectively visualize different datasets.



Instructor Demonstration

GeoJSON Review

What is GeoJSON?

➔ Review

GeoJSON

- GeoJSON is a geospatial data interchange format based on JavaScript Object Notation (JSON). It defines several types of JSON objects and the manner in which they are combined to represent data about geographic features, their properties, and their spatial extents.

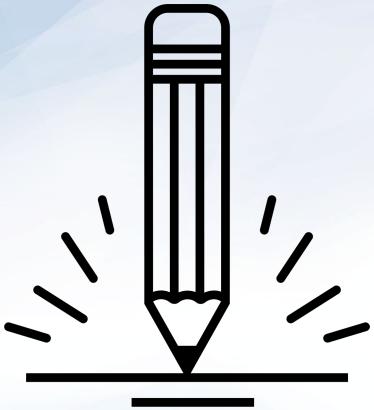
GeoJSON Review

{"type":"FeatureCollection","metadata":{"generated":1603337170000,"url":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_hour.geojson","title":"USGS All Earthquakes, Past Hour","status":200,"api":"1.10.3","count":7},"features":[{"type":"Feature","properties":{"mag":1.29,"place":"13km SW of Searles Valley, CA","time":1603335918400,"updated":1603336147381,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/ci39440911","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ci39440911.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"automatic","tsunami":0,"sig":26,"net":"ci","code":"39440911","ids":",ci39440911,","sources":",ci,","types":",nearby-cities,origin,phase-data,scitech-link,","nst":19,"dmin":0.1353,"rms":0.17,"gap":140,"magType":"ml","type":"earthquake","title":"M 1.3 - 13km SW of Searles Valley, CA"},"geometry":{"type":"Point","coordinates":[-117.5178333,35.6966667,6.45]},"id":"ci39440911"},{"type":"Feature","properties":{"mag":5.1,"place":"50 km WNW of Jiangyou, China","time":1603335819083,"updated":1603336468040,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/us6000cb4i","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/us6000cb4i.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"reviewed","tsunami":0,"sig":400,"net":"us","code":"6000cb4i","ids":",us6000cb4i,","sources":",us,","types":",origin,phase-data,","nst":null,"dmin":11.379,"rms":0.57,"gap":41,"magType":"mb","type":"earthquake","title":"M 5.1 - 50 km WNW of Jiangyou, China"},"geometry":{"type":"Point","coordinates":[104.2181,31.9295,16.96]},"id":"us6000cb4i"},{"type":"Feature","properties":{"mag":1.12,"place":"15km S of Trona, CA","time":1603334693410,"updated":1603335588520,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/ci39440895","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ci39440895.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"reviewed","tsunami":0,"sig":19,"net":"ci","code":"39440895","ids":",ci39440895,","sources":",ci,","types":",focal-mechanism,nearby-cities,origin,phase-data,scitech-link,","nst":15,"dmin":0.1147,"rms":0.15,"gap":131,"magType":"ml","type":"earthquake","title":"M 1.1 - 15km S of Trona, CA"},"geometry":{"type":"Point","coordinates":[-117.406,35.6348333,10.15]},"id":"ci39440895"},{"type":"Feature","properties":{"mag":2,"place":"15km W of Ludlow, CA","time":1603334429420,"updated":1603335569542,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/ci39440887","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ci39440887.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"reviewed","tsunami":0,"sig":62,"net":"ci","code":"39440887","ids":",ci39440887,","sources":",ci,","types":",nearby-cities,origin,phase-data,scitech-link,","nst":19,"dmin":0.1323,"rms":0.14,"gap":48,"magType":"ml","type":"earthquake","title":"M 2.0 - 15km W of Ludlow, CA"},"geometry":{"type":"Point","coordinates":[-116.3166667,34.6976667,3.3]},"id":"ci39440887"},{"type":"Feature","properties":{"mag":0.3,"place":"30 km SSE of Mina, Nevada","time":1603333972210,"updated":1603334302902,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/nn00779882","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/nn00779882.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"automatic","tsunami":0,"sig":1,"net":"nn","code":"00779882","ids":",nn00779882,","sources":",nn,","types":",origin,phase-data,","nst":10,"dmin":0.011,"rms":0.03,"gap":133.28,"magType":"ml","type":"earthquake","title":"M 0.3 - 30 km SSE of Mina, Nevada"},"geometry":{"type":"Point","coordinates":[-117.9923,38.1273,10.61]},"id":"nn00779882"},{"type":"Feature","properties":{"mag":4.7,"place":"Reykjanes Ridge","time":1603333903888,"updated":1603334600040,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/us6000cb47","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/us6000cb47.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"reviewed","tsunami":0,"sig":340,"net":"us","code":"6000cb47","ids":",us6000cb47,","sources":",us,","types":",origin,phase-data,","nst":null,"dmin":9.792,"rms":0.98,"gap":119,"magType":"mb","type":"earthquake","title":"M 4.7 - Reykjanes Ridge"},"geometry":{"type":"Point","coordinates":[-35.4046,53.0278,10]},"id":"us6000cb47"},{"type":"Feature","properties":{"mag":2,"place":"7 km NW of Fritz Creek, Alaska","time":1603333651473,"updated":1603334659397,"tz":null,"url":"https://earthquake.usgs.gov/earthquakes/eventpage/ak020dlkfgbw","detail":"https://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ak020dlkfgbw.geojson","felt":null,"cdi":null,"mmi":null,"alert":null,"status":"automatic","tsunami":0,"sig":62,"net":"ak","code":"020dlkfgbw","ids":",ak020dlkfgbw,","sources":",ak,","types":",origin,","nst":null,"dmin":null,"rms":0.85,"gap":null,"magType":"ml","type":"earthquake","title":"M 2.0 - 7 km NW of Fritz Creek, Alaska"},"geometry":{"type":"Point","coordinates":[-151.3941,59.784,82.6]},"id":"ak020dlkfgbw"}],"bbox":[


```
{
  "type": "Feature",
  "properties": {
    "mag": 0.5,
    "place": "4km W of Cobb, California",
    "time": 1476329457770,
    "updated": 1476329552105,
    "tz": -420,
    "url": "http://earthquake.usgs.gov/earthquakes/eventpage/nc72711736",
    "detail": "http://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/nc72711736.geojson",
    "felt": null,
    "cdi": null,
    "mmi": null,
    "alert": null,
    "status": "automatic",
    "tsunami": 0,
    "sig": 4,
    "net": "nc",
    "code": "72711736",
    "ids": ",nc72711736,",
    "sources": ",nc,",
    "types": ",general-link,geoserve,nearby-cities,origin,phase-data,",
    "nst": 11,
    "dmin": 0.006811,
    "rms": 0.01,
    "gap": 70,
    "magType": "md",
    "type": "earthquake",
    "title": "M 0.5 - 4km W of Cobb, California"
  },
  "geometry": {
    "type": "Point",
    "coordinates": [
      -122.7771683,
      38.8195,
      0.35
    ]
  },
  "id": "nc72711736"
},
```


GeoJSON may come with a "properties" object containing some metadata about the feature. In particular we are provided with some immediately useful information such as the place the earthquake occurred, the magnitude, and the time it was recorded.

When using GeoJSON with Leaflet, Leaflet expects each feature object to have a "geometry" property containing information about the type of marker that should be displayed and its coordinates.



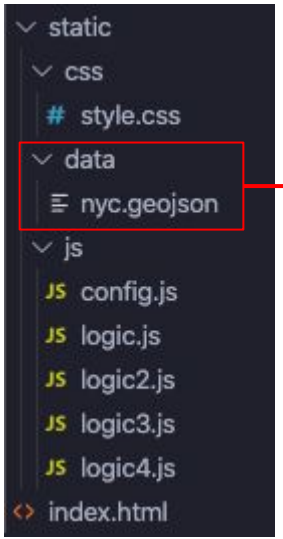
Everyone Do: NY Neighborhoods

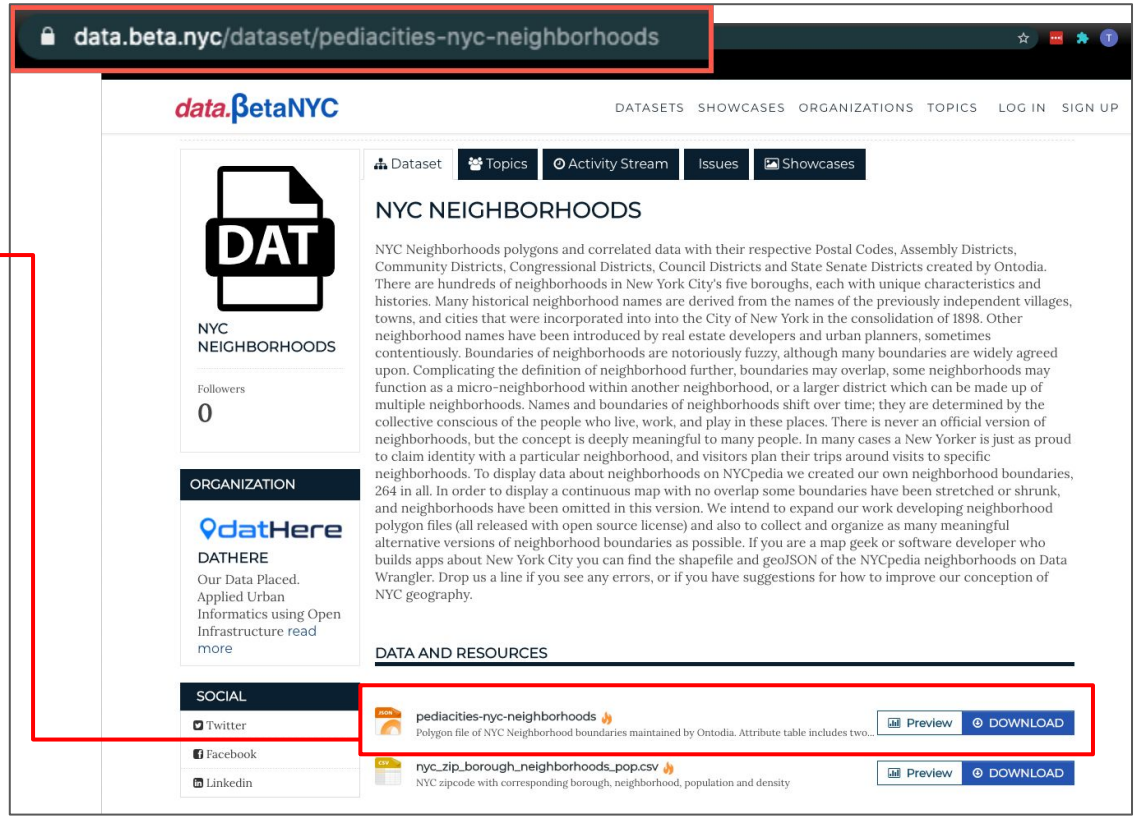
In this activity, we will be diving into some advance Leaflet/GeoJSON functionality by building a map of New York City broken down by boroughs and neighborhoods.

Suggested Time:
20 Minutes

Everyone Do: NY Neighborhoods

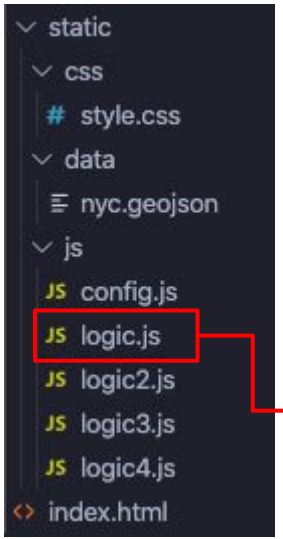
➔ File Structure

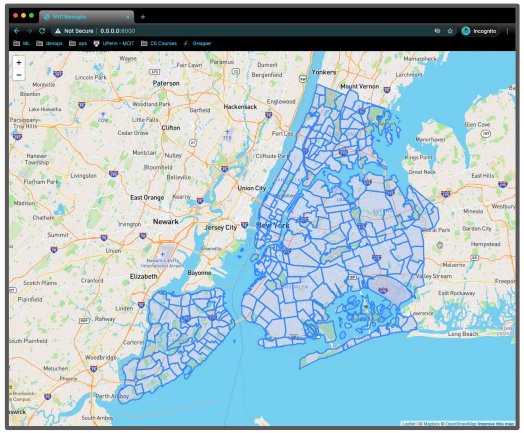




Everyone Do: NY Neighborhoods

➔ File Structure



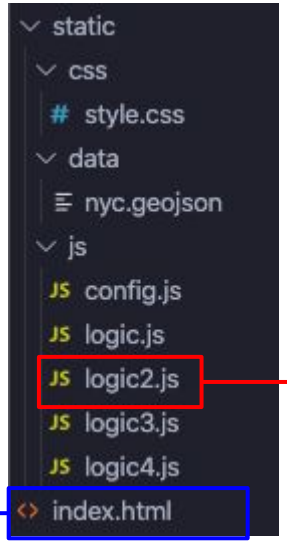


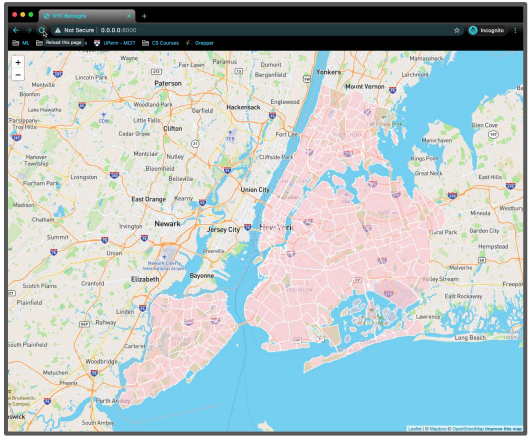
- From the command line navigate to where `index.html` is and run: `python -m http.server`.
- Open http://0.0.0.0:8000/ in a browser.

```
// Creating map object
var myMap = L.map("map", {
  center: [40.7128, -74.0059],
  zoom: 11
});

// Adding tile layer
L.tileLayer("https://api.mapbox.com/styles/v1/{id}/tiles/{z}/{x}/{y}?access_token={accessToken}", {
  attribution: "© <a href='https://www.mapbox.com/about/maps/'>Mapbox</a> © <a href='http://www.openstreetmap.org/copyright'>OpenStreetMap</a> <strong><a href='https://www.mapbox.com/map-feedbac
  tileSize: 512,
  maxZoom: 18,
  zoomOffset: -1,
  id: "mapbox/streets-v11",
  accessToken: API_KEY
}).addTo(myMap);

// Use this link to get the geojson data.
var link = "static/data/nyc.geojson";

// Grabbing our GeoJSON data..
d3.json(link, function(data) {
  // Creating a GeoJSON layer with the retrieved data
  L.geoJson(data).addTo(myMap);
});
```

Everyone Do: NY Neighborhoods

➔ File Structure





- From the command line navigate to where `index.html` is and run: `python -m http.server`.
- Open http://0.0.0.0:8000/ in a browser.

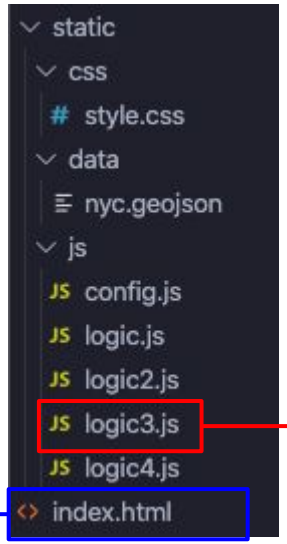
➔ In the `index.html` file change to `logic2.js`

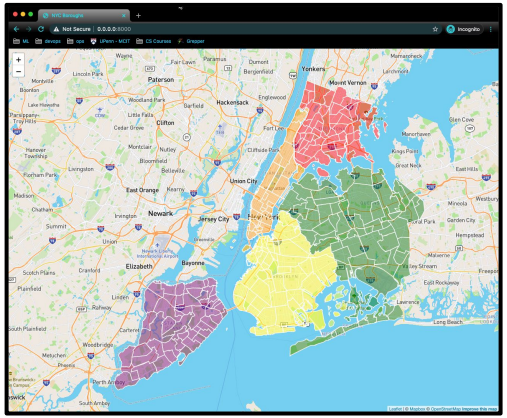
```
<!-- JS -->
<script type="text/javascript" src="static/js/logic2.js"></script>
```

```
// Creating map object
var myMap = L.map("map", {
  center: [40.7128, -74.0059],
  zoom: 11
});

// Adding tile layer
L.tileLayer("https://api.mapbox.com/styles/v1/{id}/tiles/{z}/{x}/{y}?access_token={accessToken}", {
  attribution: "© <a href='https://www.mapbox.com/about/maps/'>Mapbox</a> © <a href='http://www.openstreetmap.org/copyright'>OpenStreetMap</a> <strong><a href='https://www.mapbox.com/map-feedback
  tileSize: 512,
  maxZoom: 18,
  zoomOffset: -1,
  id: "mapbox/streets-v11",
  accessToken: API_KEY
}).addTo(myMap);

// Use this link to get the geojson data.
var link = "static/data/nyc.geojson";

// Our style object
var mapStyle = {
  color: "white",
  fillColor: "pink",
  fillOpacity: 0.5,
  weight: 1.5
};

// Grabbing our GeoJSON data..
d3.json(link, function(data) {
  // Creating a GeoJSON layer with the retrieved data
  L.geoJson(data, {
    // Passing in our style object
    style: mapStyle
  }).addTo(myMap);
});
```

Everyone Do: NY Neighborhoods

➔ File Structure

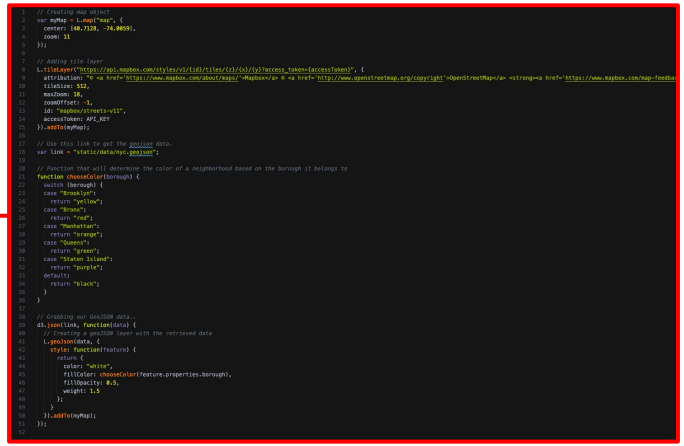




- From the command line navigate to where `index.html` is and run: `python -m http.server`.
- Open http://0.0.0.0:8000/ in a browser.

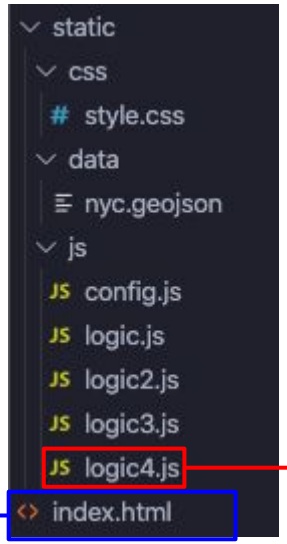
➔ In the `index.html` file change to `logic3.js`

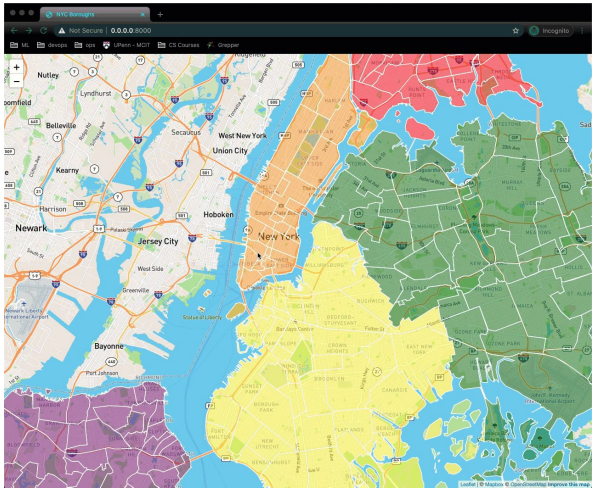
```
<!-- JS -->
<script type="text/javascript" src="static/js/logic3.js"></script>
```



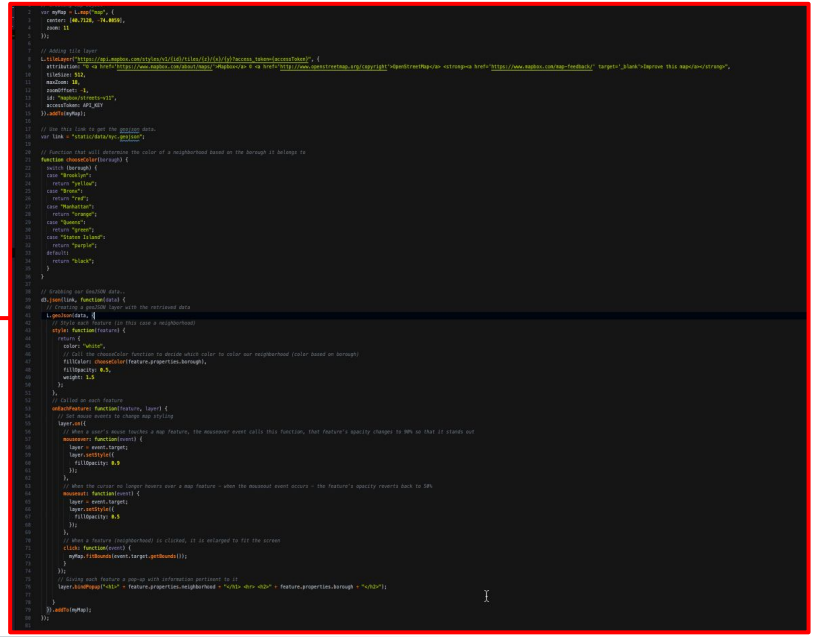
Everyone Do: NY Neighborhoods

➔ File Structure



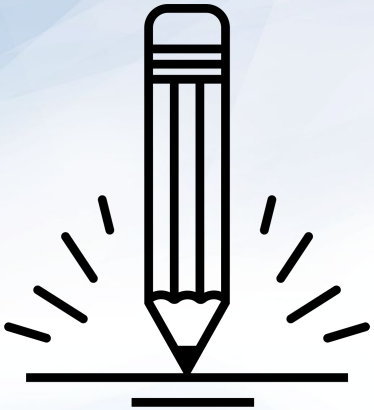


- From the command line navigate to where `index.html` is and run: `python -m http.server`.
- Open http://0.0.0.0:8000/ in a browser.



➔ In the `index.html` file change to `logic4.js`

```
<!-- JS -->
<script type="text/javascript" src="static/js/logic4.js"></script>
```



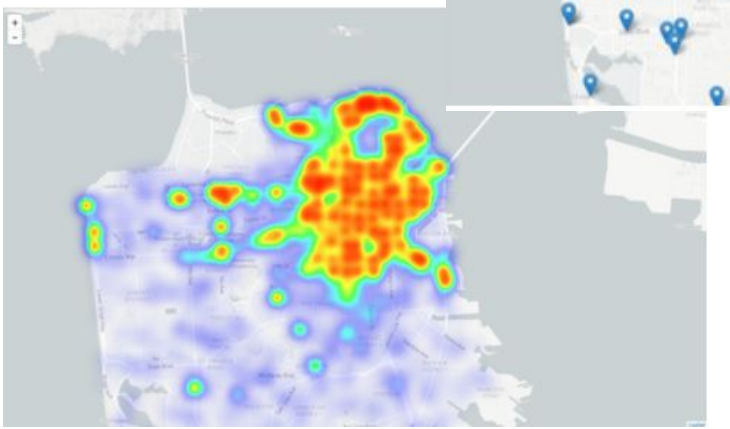
Everyone Do - Intro to Plugins: Heat Map of Crime in San Francisco

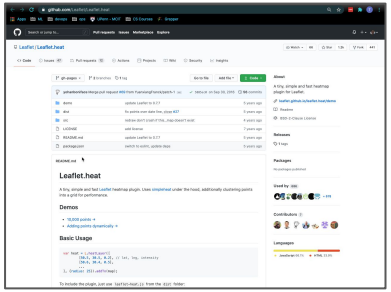
In this activity, we will focus on plotting some basic data with vanilla Leaflet and adding a third party plugin to make a really cool map!

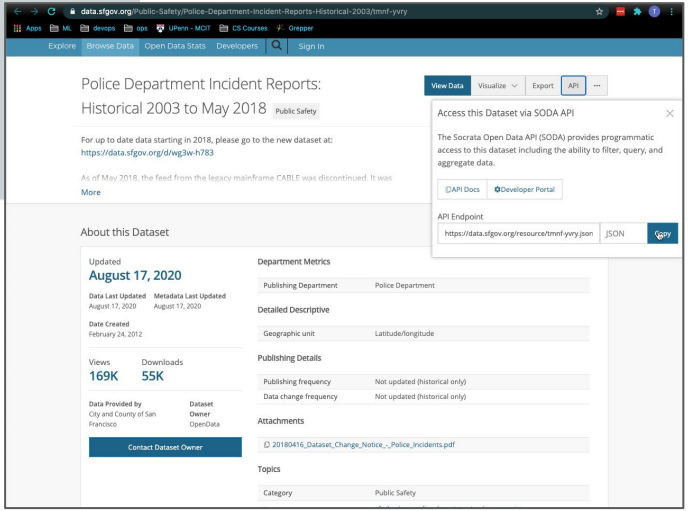
Suggested Time:
15 Minutes

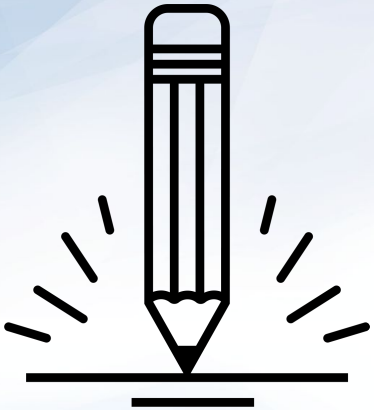
Everyone Do - Intro to Plugins: Heat Map of Crime in SF

➔ Live Code









Activity: Rat Cluster

In this activity, you will be taking data from NYC open data website and plotting it with the help of the Leaflet plugin.

Suggested Time:
30 Minutes

Activity: Quick Labeling Exercise

Instructions:

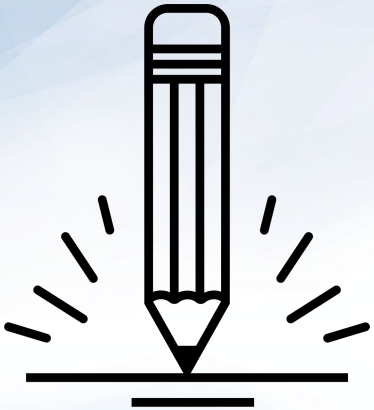
- You will getting your hands dirty visualizing rodent sightings in New York City! Gross!
- Check out the data for all 311 Service Requests in NYC (police non-emergency).
    - You are going to have to build a query URL from the above so that only rodent complaints from 2016 are returned.
    - You should limit the data returned to 10,000 data points.
- Once you have successfully plotted your rat data, work towards incorporating the Leaflet.markercluster plugin.
    - Cluster plugins can help to declutter a map with tons of data on it!
- **Hints**:
    - You can increase the data limit to 10,000 AFTER you get the cluster plugin working, but plotting 10,000 normal markers on a map may slow down your computer quite a bit.
- **Bonus**:
    - If you finish plotting rodent-sighting data on the map, use the 311 service Requests data to plot a similar graph with a different type of data.

Time’s Up! Let’s Review.

Countdown timer

15:00

(with alarm)



Partners Do: Choropleth

In this activity, you and your partner will work together to create a choropleth map that visualizes the median household income of Los Angeles and the surrounding counties.

Suggested Time:
30 Minutes

Partners Do: Choropleth

Instructions:

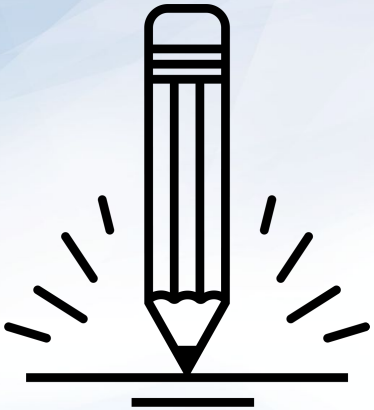
- Over the course of this activity, you and your partner will be creating a choropleth map which will visualize the median household incomes of LA and surrounding counties.
    - A choropleth map is one in which areas are shaded or patterned in proportion to the statistical variable being represented.
    - The choropleth map provides an easy way to visualize how a measurement varies across a geographic area, showing the level of variability within a region.
- You and your partner will be using a new plugin called Leaflet-Choropleth to create this map which you can find HERE in the "dist" folder of the repository.
- You will be working your way through this activity step-by-step with your partner and the class will reconvene after each step has been accomplished in order to review.
- **Hints**:
    - You can increase the data limit to 10,000 AFTER you get the cluster plugin working, but plotting 10,000 normal markers on a map may slow down your computer quite a bit.
    - The colorbrewer2 website provides color schemes (in hex values) that you can use to customize a choropleth map.

Partners Do: Choropleth

Individual Steps:

- Step 1: Grab all of the data with d3 and plot it on the map.
- Step 2: Download the Leaflet-Choropleth repository, `choropleth.js`, place it in your js folder, and uncomment the `<script type="text/javascript" src="static/js/choropleth.js"></script>` in your `index.html` file.
- Step 3: Using the Leaflet-Choropleth documentation, create a new choropleth layer.
  - Make sure to change the `valueProperty` to the property that we wish our map to be based on.
  - Define an `onEachFeature` method that binds a popup containing the value of the feature to the layer.
- Step 4: Consult the examples and Leaflet documentation on how to add a legend.
  - Use `L.control` to add a control (and choose its position).
  - Use `L.DomUtil.create('div', 'info legend')` to create a div with the classes `info` & `legend`.
  - Loop through the colors and values of your choropleth data and add them with `div.innerHTML`.
  - Return div when done.

Time’s Up! Let’s Review.



Groups Do: A Map of Your Very Own

In this activity, you and your group will create a map from scratch.

Suggested Time:
30 Minutes

Everyone Do: Mini-Presentations on Maps